

# Re: JVM/Java memory footprint

---

*Source:* <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2007-01/msg03141.html>

---

- *From:* [nukleus@xxxxxxxxxxxxx](mailto:nukleus@xxxxxxxxxxxxx) (nukleus)
  - *Date:* Tue, 30 Jan 2007 10:54:25 GMT
- 

In article <1170107776.767644.56740@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>, "Daniel Pitts" <googlegroupie@xxxxxxxxxxxxx> wrote:

On Jan 29, 6:42 am, alex...@xxxxxxxxxx wrote:

Hi,  
Though I have some three four years experience in C++ ,I did not have that much oppurtunity to work in Java. Currently I was doing some analysis for a very simple CLI and was surprised to come with a memory restriction. I found that if I use Java for developing the CLI application I will be exhausting the memory of our Application Server (AS). Just to mention the architecture ,users (telecom operators) use a metaframe server client ( =something like remote desktop) to login to the AS and then open a GUI to work on it. With Java I can service only about 15 clients with the available memory . I just checked the reason for this resource crunch and found that already many Java based GUI's are served by the AS and each is taking some 25 MB or more.

I looked into this footprint issue as in my case, the amount of heap required could be very large and that is for each thread running under the same instance of an app. Just to run the simplest application, JVM swallows tens of megabytes. And when you run the application that could be in a way similar to your situation, literally hundreds of megabytes are simply eaten up. I ended up inserting code to free up all the huge lists and buffers that are generated on the fly and not to preload large buffers with the data, say news article bodies, and postpone it to the very last moment, when this data is needed. Then, once it is used, I explicitly empty the buffers by removing the vector contents or equivalent thereof.

Just to process a 25000 article archive, you could be loosing terabytes, and not even clear where, cause your entire archive is less than 50 megs. Where did all those terabytes go?  
I am still trying to figure it out.

Seemed to help in that i see that the garbage collector is cleaning

## Re: JVM/Java memory footprint

all this stuff and heap grows as lil as possible. But it places about major restriction on your code as you have to juggle things like a clown in circus, and have to let go of things that were already referenced and loaded, as you might want to do for some feature operations.

I do not recall having ANY of these problems with all my C++ experi